

Optimal Lempel-Ziv based lossy compression for memoryless data: how to make the right mistakes

Narayana P. Santhanam
Dept. of Electrical Engg,
University of Hawaii at Manoa
nsanathan@hawaii.edu

Dharmendra Modha
IBM Almaden Research Center
dmodha@almaden.ibm.com

October 19, 2012

Abstract

Please note: this document is very much work in progress till 10/31 by my estimate. If something seems off, it probably is. Please email N Santhanam above—you can get notes for clarification. Compression refers to encoding data using bits, so that the representation uses as few bits as possible. Compression could be lossless: *i.e.* encoded data can be recovered exactly from its representation) or lossy where the data is compressed more than the lossless case, but can still be recovered to within prespecified distortion metric. In this paper, we prove the optimality of *Codelet Parsing*, a quasi-linear time algorithm for lossy compression of sequences of bits that are independently and identically distributed (*iid*) and Hamming distortion. Codelet Parsing extends the lossless Lempel Ziv algorithm to the lossy case—a task that has been a focus of the source coding literature for better part of two decades now.

Given *iid* sequences \mathbf{x} , the expected length of the shortest lossy representation such that \mathbf{x} can be reconstructed to within distortion D is given by the rate distortion function, $r(D)$. We prove the optimality of the Codelet Parsing algorithm for lossy compression of memoryless bit sequences. It splits the input sequence naturally into phrases, representing each phrase by a *codelet*, a potentially distorted phrase of the same length. The codelets in the lossy representation of a length- n string \mathbf{x} have length roughly $(\log n)/r(D)$, and like the lossless Lempel Ziv algorithm, Codelet Parsing constructs codebooks logarithmic in the sequence length.

Introduction

Kac’s lemma [1] for stationary ergodic sources formalizes the connection of the *recurrence* time of events with their probabilities. This connection implies an elegant way to recursively compress sequences from stationary ergodic sources to their entropy, formalized by the Lempel Ziv algorithm for lossless compression.

The theoretical and commercial importance of the Lempel Ziv algorithm and its variants have not only been established for compression problems, but also for classification [2] and denoising [3] algorithms. In addition to their theoretical guarantees, these algorithms have attractive computational and storage properties, are often entirely data driven, and do not rest on sensitive choices of parameter values. It is thus not surprising that Lempel Ziv based algorithms form the core of compression algorithm software, including WINZIP, **gzip**, and the UNIX **compress** algorithms. Additionally, Lempel Ziv compression has had profound influence in the study of complexity, see for example, [4, 5]. For many researchers, this angle perhaps outweighs even the commercial significance of Lempel Ziv compressors.

Lossy compression

Surprisingly, no algorithms as attractive and simple as the Lempel Ziv algorithm are known for lossy compression. In fact, in the recent past, some researchers were pessimistic about the problem in general, see [6] for details. For example, [7, p. 2709] noted that “All universal lossy coding schemes found to date lack the relative simplicity that imbues Lempel-Ziv coders and arithmetic coders with economic viability”.

Of course, a lot of research continues on lossy source compression algorithms, mainly with an eye on the potential theoretical and practical benefits of having such algorithms.

Prior work

We present a representative, but necessarily brief and non-exhaustive review of various known lossy coding schemes, focussing on algorithmic results. For references to earlier results on existence of universal lossy codes involving exponential-time constructions, see, Kieffer [8]. We confine our discussion here to finite discrete source and reproduction alphabets; for an extensive survey of results for real-valued sources, see [9]. Among these, we are particularly interested in papers that have focussed on lossy extensions of the Lempel-Ziv algorithm.

Most algorithms have naturally used approximate string matching [10, 11] instead of exact string matching as in the Lempel-Ziv algorithms. The unresolved question has always been which of the “approximately matching” representations to choose. Cheung and Wei [12] extended a *move-to-front* algorithm to lossy source coding. The algorithm is sup-optimal [13]. Later, Zhang and Wei [14] proposed an universal, on-line lossy coding algorithm for the fixed-rate case. Morita and Kobayashi [15] extended the LZW algorithm, but their algorithm is known to be sub-optimal for memoryless sources [13]. Constantinescu and Storer [16, 17] combined ideas from lossless Lempel-Ziv algorithms and vector quantization to design first *practical* implementations of lossy image compression based on approximate string matching. The problem of “selecting amongst multiple matches” mentioned above was termed the “Match Heuristic” in their work; see, also, Storer [18, p. 111]. Steinberg and Gutman [19] and Luczak and Szpankowski [20] considered the fixed-database version of the Lempel-Ziv algorithm, and provided sub-optimal performance guarantees. However, Yang and Kieffer [13] established that all previous fixed-database extensions of the Lempel-Ziv algorithm are suboptimal.

Kontoyiannis [21] presented a scheme where multiple databases are used at the encoder, which must also be known to the decoder. However, when the reproduction alphabet is large, the number of training databases is unreasonably large. Atallah et al. [22] considered a cubic-time, adaptive algorithm (PMIC) in the spirit of LZ77. Their algorithm is not sequential in the sense of [23], since its encoding delay grows faster than $o(n)$. Alzina et al. [24] combined ideas from [22] and [16, 17] to propose a 2D-PMIC algorithm that is more suited for two dimensional images.

Continuing the quest for Lempel-Ziv-type lossy algorithms, Zamir and Rose [25] further studied the algorithm in [15]. From the multiple codewords that may match a source word, they suggest choosing one “at random”. From a theoretical perspective, by assuming uniqueness, Zamir and Rose [26] proposed a natural type selection scheme for finding the type of the optimal reproduction distribution. In later work, Kochman and Zamir [27] pointed out that the theoretical procedure in [26] is in itself not practical and demonstrated an application of natural-type selection to on-line codebook selection from a parametric class. Along a different line, Yang and Kieffer [28] have proposed exponential-time Lempel-Ziv-type block codes that are universal (for stationary, ergodic sources and for individual sequences). In a related work, Yang and Zhang [29] presented fixed-slope universal lossy coding schemes that search for the reproduction sequence through a trellis in a fashion reminiscent of the

Viterbi algorithm.

The lossy coding problem has been approached using methods fundamentally different from the Lempel Ziv like approaches as well. Matsunaga and Yamamoto [30] considered LDPC codes for lossy data compression. In this line of work, Wainwright and Maneva [31] looked at message passing and Low Density Generator matrices (LDGM), while Martinian and Wainwright [32] looked into the construction of LDGMs and compound code constructions, showing the existence of compound LDGM-LDPC constructions that achieve the rate-distortion bound. Further bounds on the performance of these constructions have been considered in [33]. In another line of attack, Jalali, Montanari and Weissman approach the problem using dynamic programming approaches [34].

Challenges

In this paper, we consider lossy encoding of memoryless data. What constitutes progress at a conceptual level? The algorithm we consider, Codelet Parsing, reduces to the Lempel Ziv algorithm (LZ78 version) for lossless encoding, and we believe that Codelet Parsing may be optimal for stationary ergodic sources as well.

One way to think of lossy encoding is as follows. We construct a *codebook* \mathcal{C} , a set of sequences substantially smaller than the set of all possible sequences. Given any sequence \mathbf{x} , we fix an element of \mathcal{C} as its representation. Thus, for any sequence \mathbf{x} , we only have to describe which element in \mathcal{C} it maps to (rather than all possible sequences). If \mathcal{C} has been chosen well, every sequence has some sequence of \mathcal{C} that is fairly close to it. Thus the crux of the lossy compression problem is (i) to construct \mathcal{C} , and (ii) to search for a representation. The minimum size of \mathcal{C} is characterized through the rate distortion function $r(D)$.

We sketch a rough picture of the problem of lossy compression now. While not necessary for the results of our paper, most of the statements below can be made formal. If a length- n sequence \mathbf{X} is generated *iid* Bernoulli p , the probability \mathbf{X} matches a length- n sequence \mathbf{y} to within distortion D is highest if the type of \mathbf{y} is $(p - D)/(1 - 2D)$. The probability of match is then $2^{-nr(D)}/\text{poly}(n)$. Thus if we are to encode length n sequences, $|\mathcal{C}| \geq \text{poly}(n)2^{nr(D)}$ in order to satisfy the distortion budget D . In fact, a randomly chosen \mathcal{C} from sequences with type $(p - D)/(1 - 2D)$ will cover almost all input sequences with size $|\mathcal{C}| = \text{poly}(n)2^{nr(D)}$. Thus *random coding* uses $\geq nr(D) + \mathcal{O}(\log n)$ bits to represent a string. This approach is clearly not practical (both construction and search take exponential time) and we look for more efficient ways to achieve the goal by using more structured codebooks.

Lempel Ziv approaches circumvent the problem of exponential encoding and search time with a recursive construction. Rather than construct codebooks for length n sequences, one constructs a set \mathcal{D} of sequences of length $\frac{(\log n)}{r(D)}$. Often, codebooks over lengths smaller than the sequence length are referred to as *dictionaries* in Lempel-Ziv literature to avoid confusion, and we adopt the same convention. The algorithm splits the length- n sequence \mathbf{X} into *phrases* of length $\frac{(\log n)}{r(D)}$, representing each phrase by one of the elements of \mathcal{D} . The strength of this approach is that the construction of \mathcal{D} happens naturally using just the data to be encoded, and is known to capture the probability laws governing the data as long as the data is stationary ergodic (not just memoryless).

Furthermore, a simple argument about recurrence time of events shows that it is not possible to estimate probabilities of all strings of length $\Omega(\log n)$ using n samples—a fact that will come into play if the algorithms are to be extended for all stationary ergodic sources. Thus, the dictionaries cannot be over sequences longer than $\mathcal{O}(\log n)$ if we have the goal of extending our algorithm to all stationary ergodic sources.

What should we expect from all this? We should expect an approach using the Lempel Ziv theme to have redundancy (the excess bits over the rate distortion $nr(D)$ term) commensurate with random

encoding of sequences of length $(\log n)/r(D)$. Comparing with the numbers given above for random encoding of length n sequences, we conclude that such approaches use $nr(D) + \mathcal{O}(\frac{n \log \log n}{\log n})$ for length n sequences. However, the complexity of search through \mathcal{D} to represent any phrase of length $(\log n)/r(D)$ is linear in n , leading to an overall complexity of $\mathcal{O}(n^2/(\log n))$ in order to encode a sequence of length n .

Note that actually adapting the Lempel Ziv theme is non-trivial. In particular, how does one guarantee that the dictionary \mathcal{D} constructed does match the performance of a randomly chosen and *good* codebook of length $(\log n)/r(D)$? This is analogous to the channel coding problem for communication, where a randomly chosen code is good with high probability—yet constructing practical codes that are optimal took almost 60 years of intense research. Indeed, the connections run deeper—lossy compression is a *covering* problem, while channel coding is a *packing* problem.

Here we show that Codelet Parsing built on the Lempel Ziv theme has a redundancy of $\mathcal{O}(\frac{\log \log n}{\log n})$ as expected. However, Codelet Parsing constructs the dictionary \mathcal{D} in a more structured manner than brute force random construction, and finding a match requires only $\text{poly}(\log n)$ (not linear) complexity on an average. Thus Codelet Parsing is a quasi *linear* algorithm. At the level of encoding length- n sequences, this is seemingly only an improvement from quadratic to linear complexity (notwithstanding the fact that it is not even clear how to achieve quadratic complexity), but such an improvement also indicates a new way to build the dictionary.

Contributions

This paper builds on the Lempel Ziv approach along the lines of [35, 36, 6]. In particular, we analyze an idealization of a Lempel Ziv like algorithm called Codelet Parsing, proposed by the authors in [37]. In a preliminary paper [6], we showed convergence of Codelet Parsing’s coding rate (the number of bits used to describe the lossy representation of a string, normalized by the length of the string) to the rate distortion function, when the input string is *iid* and the distortion is fixed to be Hamming distortion.

In this paper, we obtain a covering lemma that allows us to characterize the rate of convergence of the coding rate as $\mathcal{O}(\frac{\log \log n}{\log n})$ (exponentially better than the loose estimate in [6]). It is important to highlight how this result substantially strengthens [6].

In particular, we note a few important points. The distorted phrases are of length roughly $(\log n)/r(D)$ and are obtained by searching through a codebook (maintained as a complete binary tree as in the LZ78 setup).

1. The sequences in this codebook are not obtained by exhaustive search. Instead, they are recursively obtained by calling on codebook constructions over shorter lengths of length $\mathcal{O}(\log \log n)$. In addition, searching for an approximate match does not require an exhaustive search over sequences of length $(\log n)/r(D)$.
2. The shorter codebook constructions work in synergy in a manner of speaking since convergence to $r(D)$ is $\mathcal{O}(\frac{\log \log n}{\log n})$. This rate is almost what we should expect even for exhaustive codebook constructions of length $\mathcal{O}(\log n)$.

A consequence of the first point is that we obtain an algorithm that is quasi-*linear* (linear with log factors) complexity. This is a savings from the potentially super-quadratic complexity if we exhaustively construct or search through codebooks of length $\log n/r(D)$,

To put the second point in perspective, the convergence rate of our algorithm is exponentially faster than what could have been obtained by partitioning \mathbf{x} into phrases of length $\mathcal{O}(\log \log n)$, and representing each phrase in a lossy manner using a codebook of length $\mathcal{O}(\log \log n)$.

1 Preliminaries and combinatorial interpretations

1.1 Rate-Distortion and Lower-Mutual-Information

Let $X^n = X_1, X_2, \dots$, where $X_i \in \{0, 1\}$ for all i , be a realization of an *iid* process P , with the marginal distribution on X_i being $P(X_i = 1) = p$. We represent a string of length n , X^n using a potentially distorted $Y^n \in \{0, 1\}^n$. Let $d(X^n, Y^n)$ denote the Hamming distortion between X^n and Y^n . We adhere to an expected distortion constraint, namely $\mathbb{E}d(X^n, Y^n) < D$. It is customary to call Y^n the *codeword* used for the lossy representation of X^n . Note that Y^n is not necessarily *iid* and is determined by the algorithm used to pick codewords.

The *rate distortion* function captures, asymptotically, the minimum number of bits that have to be used to describe strings of length n to within distortion D . Interestingly, it has a *single letter* characterization, meaning that it can be specified by looking at the joint distribution over a pair of bits (Y, X) such that $P(X = 1) = p$. The conditional distributions on X given Y correspond to a *channel*, while Y is interpreted as the channel input and X the channel output.

Let \mathcal{W} be the set of all possible channels. The *rate-distortion* function is

$$r(D) = R(P, D) = \min_{\substack{q', \omega \in \mathcal{W}: Y \sim q', X \sim p \\ \mathbb{E}d(X, Y) \leq D}} I(X, Y)$$

where $I(X, Y)$ denotes the *mutual information* and $Y \sim q'$ means $P(Y = 1) = q'$.

The lossy coding problem is essentially a covering problem. Suppose we consider length- n sequences X^n generated by an *iid* measure P , satisfying $P(X_i = 1) = p$ (as before). Say we want the probability of length n sequences of type p that are within distortion D from a sequence \bar{y} with type q . This probability again has a single letter characterization in terms of a pair of binary variables (Y, X) , where $Y \sim q$ and $X \sim p$. In particular, we define

$$I_m(q, p, D) \stackrel{\text{def}}{=} \min_{\substack{\omega \in \mathcal{W}: X \sim p, Y \sim q \\ d(q, \omega) \leq D}} I(X, Y),$$

where we are minimizing the mutual information $I(X, Y)$ over all joint distributions consistent with the marginals being $X \sim p$ and $Y \sim q$, and $\mathbb{E}d(X, Y) \leq D$. The probability we want is then $2^{-nI_m(q, p, D) + \mathcal{O}(\log n)}$. $I_m(q, p, D)$ is a convex function of q for a fixed p , with a minimum at the *optimal reproduction type* q^* .

Intuitively speaking, codewords with the optimal reproduction type have the largest D -balls among sequences of type P , hence, yield the best covering. For a precise formulation of the above concepts, see [14, 26]. However, to just obtain the estimates given above, a simple combinatorial calculation followed by picking the dominant term suffices.

1.2 Ballot box problem

We have an expected distortion constraint between a sequence X^n generated by P and its codeword Y^n . As we will see, we obtain Y^n by first breaking X^n into disjoint phrases $X^n = \mathbf{X}^{(1)}, \dots, \mathbf{X}^{(r)}$ (where $r = \mathcal{O}(n/\log n)$), and representing each phrase $\mathbf{X}^{(i)}$ by a *codelet* $\mathbf{y}^{(i)}$ of the same length, such

that $d(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \leq D$. Such an approach however leads to lack of sufficient structure in the codebooks generated, leading to quadratic complexity for the algorithm.

To better implement search and representation among codelets, we impose a more restrictive constraint in picking codelets. We will require not only that $d(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \leq D$ in the example above, but that every prefix of $\mathbf{x}^{(i)}$ be within distortion D of the corresponding prefix of $\mathbf{y}^{(i)}$. Namely, for any l , if \mathbf{x}' and \mathbf{y}' are l -length prefixes of $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ respectively, we require that $d(\mathbf{x}', \mathbf{y}') \leq D$ as well. We then write $\mathbf{x}^{(i)} \sim \mathbf{y}^{(i)}$ and say that $\mathbf{x}^{(i)}$ *matches* $\mathbf{y}^{(i)}$.

The important thing is that the probability that a codelet \mathbf{y} finds a match is essentially the probability of all sequences with distortion D from \mathbf{y} . In fact

(maybe state stronger too?)

Lemma 1. Let length n sequences \mathbf{X} be generated by an *iid* source P , and let the type of \mathbf{y} be q , the optimal reproduction type for P and the distortion metric D . Then

$$P(\mathbf{X} \sim \mathbf{y}) \geq \frac{(1 - D/2)^2}{n} P(B(\mathbf{y}, d)),$$

where, $\mathbf{X} \sim \mathbf{y}$ is as defined in text preceding this Lemma.

Proof We adapt a so-called *Cycle Lemma* in Dvoretzky and Motzkin [38] that has been rediscovered several times [39] in literature.

Consider sequences \mathbf{y}_0 and \mathbf{y}_1 corresponding to the zeros and ones of \mathbf{y} . We first look for sequences \mathbf{x}_0 and \mathbf{x}_1 satisfying $d(\mathbf{x}_0, \mathbf{y}_0) \leq D$ and $d(\mathbf{x}_1, \mathbf{y}_1) \leq D$, and make a sequence \mathbf{x} by replacing the zeros of \mathbf{y} with \mathbf{x}_0 and the ones of \mathbf{y} with \mathbf{x}_1 . Let \mathcal{B} be the set of all such sequences \mathbf{x} .

Suppose (\mathbf{x}_0) and (\mathbf{x}_1) are cyclic shifts of some valid \mathbf{x}_0 and \mathbf{x}_1 respectively. Then the cycle lemma of [38] states that at least $(1 - D/2)$ fraction of these cyclic shifts are $\sim \mathbf{y}_0$ and $\sim \mathbf{y}_1$ respectively—we call them *good* shifts. Note that if we replace both \mathbf{y}_0 and \mathbf{y}_1 with good shifts of \mathbf{x}_0 and \mathbf{x}_1 to obtain a sequence \mathbf{x} , then it follows that $\mathbf{x} \sim \mathbf{y}$. In addition, all sequences formed by replacing the zeros and ones with (good or otherwise) shifts of \mathbf{x}_0 and \mathbf{x}_1 have the same type, and hence the same probability under P . Thus

$$P(\mathbf{X} \sim \mathbf{y}) \geq (1 - D/2)^2 P(\mathcal{B}).$$

Furthermore, it is easy to verify that if the type of \mathbf{y} is the optimal reproduction type, (remove and use only previous equation—the next equation is unnecessary and never used)

$$P(\mathcal{B}) \geq \frac{1}{n} P(B(\mathbf{y}, D)). \quad \square$$

2 Codelet parsing

At the core of the paper is the *Codelet parsing* algorithm for lossy compression with a Hamming distortion constraint. When no distortion is allowed, the algorithm reduces to the lossless Lempel Ziv algorithm. Codelet Parsing *sequentially* parses the source sequence into non-overlapping phrases, mapping each phrase to a *codelet* in a *dictionary*. The dictionary in turn is updated.

At the block level, the codelet parsing algorithm maps a source sequence x_1^n to a distorted sequence y_1^n , and then encodes and transmits the latter without loss using a LZ78 encoder. We describe the algorithm with an example, full details are available in [37].

Example 1. Consider the string $x_1^{13} = 0110101101000$, which we will encode with allowable hamming distortion $D \leq 1/2$. We initialize a codebook $\mathcal{C}_0 = \{0, 1\}$, call the members of the codebook as

codelets, and denote the type of a string v by $\tau(v)$. At each step, we choose a codelet to represent a portion of the unparsed string, such that the codelet is within distortion $1/2$ from a matching length prefix of the unparsed string.

At step $t = 1$, the unparsed string is 0110101101000. The codelet 0 has a prefix (0) within distortion 0, while the codelet 1 does not match any prefix to within distortion $1/2$. The first bit of x_1^{13} is represented by the codelet 0, and the matching codelet 0 in \mathcal{C}_0 is replaced by its one bit extensions, namely 00 and 01, to yield \mathcal{C}_1 .

Now $\mathcal{C}_1 = \{00, 01, 1\}$, and the unparsed segment of the string is 110101101000. Note that codelet 1 has a prefix (1) within distortion 0 while the codelet 01 has a prefix (11) within distortion $1/2$. We have two choices: represent the first bit of the unparsed segment with the codelet 1, or the first two bits of the unparsed segment with 01.

To decide, we build the set of matching codelets $\mathcal{M}_1 = \{01, 1\}$. To each codelet $m \in \mathcal{M}_1$, associate the prefix r of x_1^{13} that will be parsed thus far if m is chosen, and compute the metric $I_m(\tau(m), \tau(r), D)$. Therefore for $m = 01$, the prefix r of x_1^{13} associated is 011 (0 from the first round, and 11 from this round). The metric for the codelet 01 is then $I_m(\tau(01), \tau(011), 1/2)$. Choose the codelet with the minimum metric, and update the codebook by replacing the chosen codelet with its one bit extensions. Suppose the chosen codelet is 01, $\mathcal{C}_2 = \{00, 010, 011, 1\}$, and the bits 11 are represented by 01 in this round. The unparsed string for the next round is then 0101101000. \square

As we saw in the second round above, there are usually multiple ways to parse the incoming source string and map it into codewords. Indeed the crux of the algorithm is the answer to:

How do we select between multiple parsings?

Interestingly, the most natural extension of Lempel Ziv algorithm to the lossy case—picking one of the longest codelet among the matches—is proven suboptimal in [20], in a specific LZ77 setting.

3 Idealization of codelet parsing

To understand the codelet parsing algorithm described above, we idealize the codelet parsing algorithm in order to isolate the core phenomena underlying the algorithm, and to make it amenable to a simple analysis.

(remove, add universal section) For the sake of simplicity, and because we are only analyzing the *iid* case in this paper, we assume that the Idealized Codelet Parsing algorithm knows the underlying statistics of the data. Note that in the *iid* case, we learn the underlying statistics at the rate of $\mathcal{O}(1/\sqrt{s})$, where s is the length of the string we have observed thus far, and hence at an exponentially faster rate than we would expect for any LZ type algorithm.

Modifications

Known horizon First, we assume that the blocklength of the input string \bar{x} is known in advance. Note that while this aids analysis, it is not a stringent restriction. In practice, a modification of the *doubling trick* ([40], Chapter 2.3) can be used to handle strings whose length is unknown, with asymptotically no degradation in performance. For details, please see [6].

Let \mathbf{y} be a length- L sequence with the optimal reproduction type, and let

$$\mathbf{p}_L = P(\mathbf{X} \sim \mathbf{y}),$$

where \mathbf{X} is a sequence generated by P . Now let

$$M_L = L^2/\mathbf{p}_L.$$

Further, denote an input sequence \mathbf{z} of length ℓ to be ϵ -typical if $|h(p) + \log P(\mathbf{z})| \leq \ell\epsilon$, and let $T_{\mathbf{X}}^{\ell,\epsilon}$ be the set of all ℓ -length ϵ -typical sequences.

Updating the dictionary The Idealized Codelet Parsing algorithm initializes the dictionary with all 2^ℓ ℓ -length sequences. Among them, it first obtains a set \mathcal{D}_ℓ of M_ℓ codelets of length ℓ . Then, every sequence in \mathcal{D}_ℓ is replaced with all its 2^ℓ ℓ -bit extensions, and among them $M_{2\ell}$ length- 2ℓ codelets are chosen to obtain $\mathcal{D}_{2\ell}$. The algorithm proceeds by then updating the dictionary with longer codelets, forming in turn, the sets $\mathcal{D}_{k\ell}$ for increasing values of k .

Selecting codelets by partial matching To pick any codelet to represent a portion of the unparsed, input sequence, the algorithm finds the longest matching codelet from the leaves of the dictionary tree.

Note that we can exploit because we map any codelet \mathbf{y} to only sequences \mathbf{x} such that $\mathbf{x} \sim \mathbf{y}$, finding the longest match does *not* require exhaustive search among the codelets with high probability. Following is an algorithm that does the search among L -length codelets in $\mathcal{O}(2^\ell L^2)$ operations with high probability.

Let $\mathbf{x} = x_1, x_2 \dots$ be the unparsed segment of the input.

$$Z_\ell = \{\mathbf{y} \in \mathcal{D}_\ell : \mathbf{y} \sim x_1^\ell\}.$$

be the partial matches at level ℓ . Among all the descendants of Z_ℓ in $\mathcal{D}_{2\ell}$, find all partial matches for $x_1^{2\ell}$ to obtain $Z_{2\ell}$. The crucial point to observe is

Property 1. If there exists $y_1^{2\ell} \in \mathcal{D}_{2\ell}$ such that $y_1^{2\ell} \sim x_1^{2\ell}$, then $y_1^\ell \sim x_1^\ell$, namely $y_1^\ell \in Z_\ell$. \square

Therefore $Z_{2\ell}$ contains all sequences in $\mathcal{D}_{2\ell}$ that $\sim x_1^{2\ell}$. We would not have this property if we simply obtained the sets Z by picking codelets that satisfied the distortion constraint alone. Combined with the Lemma ?? below that with high probability, $|Z_{k\ell}|$ grows polynomially rather than exponentially, obtaining Z_L for any L can be done polynomially in L . In the low probability event that $Z_{k\ell}$ grows faster than the Lemma bound, we simply give up.

Lemma 2. For all δ , with probability $\geq 1 - \delta$, simultaneously for all k

$$|Z_{k\ell}| \leq \frac{(k\ell)^4}{\delta}. \quad \square$$

4 Optimality of Codelet Parsing

We show that the Idealized Codelet Parsing algorithm is optimal. Let $X^n = X_1, \dots, X_n$ be generated by a binary memoryless source P , with $P(X_1 = 1) = p$. Let the *target* average Hamming distortion constraint be D . Let Y^n be the distorted representation of X^n output by the algorithm, and let $\mathcal{L}(Y^n)$ be the number of bits required to describe Y^n . Then,

Theorem 3. For the Idealized Codelet Parsing algorithm,

$$\frac{1}{n} \mathbb{E} \mathcal{L}(Y^n) \leq r(D) + \mathcal{O}\left(\frac{\log \log n}{\log n}\right),$$

and $\frac{1}{n} d(X^n, Y^n) \leq D$ \square

The expectation above is taken over all the choices made by the algorithm and over the input sequences.

Analysis of the cover

We first establish that the codelets provide a good cover for the source phrases.

The algorithm chooses codelets of lengths $\ell, 2\ell$ and so on. We will often refer to the length of codelets as their *depth*, since they are either internal nodes or leaves of the dictionary tree. Let \mathbf{Y}_i^L be the i 'th (in sequence) codelet chosen at depth L of the dictionary tree. Note that the dictionary is itself random (dictated by \mathbf{X} and the random choices made while populating it), and we denote by \mathcal{D}_L the dictionary at depth L once the algorithm has processed a length n sequence. For any sequence \mathbf{X} with length L , let $\mathcal{T}_L(\mathbf{X})$ be the number of codelets in \mathcal{D}_L that are within the distortion budget from \mathbf{X} . We will drop the argument of \mathcal{T}_L when writing expectations for simplicity. All expectations that follow are over \mathbf{X} and \mathcal{D} .

As mentioned before, too many matches is a sign of suboptimality. To quantify this, we compute $\mathbb{E}\mathcal{T}_L$ and $\mathbb{E}\mathcal{T}_L^2$. Together, they provide a lower bound on the probability $\mathcal{T}_L > 0$, namely the probability that \mathbf{X} is covered by some element of the dictionary at depth L .

Clearly $\mathbb{E}\mathcal{T}_L$ is easy to compute for any L by linearity of expectation. However $\mathbb{E}\mathcal{T}_L^2$ is somewhat trickier to bound, but is well behaved. We show in Lemma 8 that when averaged over all possible codebooks, $\mathbb{E}\mathcal{T}_L^2$ is lower than the corresponding expectation if we chose M_L codelets at random. From [], random choice of codelets leads to good covers with overwhelming probability. We will therefore conclude that, the cover gets better as we parse longer. Computation of $\mathbb{E}\mathcal{T}_L^2$ is somewhat involved, but the algebra is simplified for a Bernoulli $1/2$ source.

We first note that the codebook construction contains symmetries that we will need to exploit for Lemma 8.

Lemma 4. Let $\mathcal{T}_L = \binom{L}{Lq}$. For all $y \in T_q^L$,

$$\mathbb{P}(y \in \mathcal{D}_L) = \frac{M_L}{\mathcal{T}_L}$$

Proof Suppose the length of y be $L = k\ell$ and let $y' = y'_1{}^\ell, y'_{\ell+1}{}^{2\ell}, \dots, y'_{(k-1)\ell+1}{}^L$. Note that each $y'_{i\ell}{}^{(i+1)\ell}$ can be obtained from the corresponding subsequence $y_{i\ell+1}{}^{(i+1)\ell}$ by some permutation of bit locations of the later, since both bit sequences have the same type. Represent these permutations by $\sigma_0, \dots, \sigma_{k-1}$, and we write $y'_1{}^\ell = \sigma_0(y_1{}^\ell)$ as a shorthand. These permutations are not unique, however we will fix one valid value for each of $\sigma_0, \dots, \sigma_{k-1}$.

Let $XC(y)$ be the set of length n input sequences and the corresponding choices between multiple matches made by the algorithm that induce $y \in \mathcal{D}_L$. Corresponding to each input sequence \mathbf{x} that could induce y , we represent the choices as numbers, one for each phrase, indicating (in lexicographic order) which of the codelets that $\sim \mathbf{x}$ are chosen. Thus,

$$XC(y) = \{(\mathbf{x}, c) : \text{choices } c \text{ on sequence } \mathbf{x} \text{ induce } y\}.$$

Similarly for $XC(y')$.

To see that there is a bijection between $XC(y')$ and $XC(y)$, take an element $(\mathbf{x}, c) \in X(y)$. From \mathbf{x} , we obtain $\mathbf{x}' \in XC(y')$ by manipulating each phrase obtained in the parsing of \mathbf{x} . Suppose $z = z_1{}^\ell \dots z_{(m-1)\ell+1}{}^{m\ell}$ is a phrase obtained during the parsing of \mathbf{x} . If $m \leq k$ we replace z with

$$z' = \sigma_0(z_1{}^\ell) \dots \sigma_{m-1}(z_{(m-1)\ell+1}{}^{m\ell})$$

and if $m > k$ we replace z with

$$z' = \sigma_0(z_1^\ell) \dots \sigma_{k-1}(z_{(k-1)\ell}^{k\ell}) z_{k\ell+1}^{(k+1)\ell} \dots z_{(m-1)\ell+1}^{m\ell}.$$

Now to make choices among competing matches, instead of lexicographic ordering, we use the lexicographic ordering under $\prod \sigma_i^{-1}(z_{(i-1)\ell}^{i\ell})$ (replace \prod with concatenation symbol). Now, note that if (\mathbf{x}, c) yielded y , (\mathbf{x}', c) will yield y' . Finally, since *iid* probabilities of sequences do not change when their bit locations are permuted, it follows that

$$\mathbb{P}(y \in \mathcal{D}_L) = \mathbb{P}(XC(y)) = \mathbb{P}(XC(y')) = \mathbb{P}(y' \in \mathcal{D}_L). \quad \square$$

Lemma 5. Let $y_1, y_2 \in T_q^L$ be identical in the first r ℓ -length segments. Then,

$$\mathbb{P}(y_1 \text{ and } y_2 \in \mathcal{D}_L) \leq \frac{M_L}{\mathcal{T}_L} \frac{M_L}{M_{r\ell} \mathcal{T}_{L-r\ell}}. \quad \square$$

The next Lemma would easily follow from the linearity of expectation, but we provide a slightly more convoluted proof using the above Lemma 4. Let $N_{L,\mathcal{D}}(\mathbf{X})$ be the number of codelets that match \mathbf{X} in the randomly chosen codebook \mathcal{D} . For the codelet parsing algorithm described above,

Lemma 6. $\mathbb{E}N_{L,\mathcal{D}} = M_L \mathbf{p}_L$.

Proof Note that

$$\begin{aligned} \mathbb{E}N_{L,\mathcal{D}} &= \sum_{\mathbf{x}} P(\mathbf{x}) \sum_y 1(y \in \mathcal{D}_L \text{ and } y \sim \mathbf{x}) = \sum_y \mathbb{P}(y \in \mathcal{D}_L) \sum_{\mathbf{x} \in B(y,D)} P(\mathbf{x}) \\ &\stackrel{(a)}{=} \sum_y \frac{M_L}{\mathcal{T}_L} \mathbb{P}(B(y, d)) = M_L \mathbf{p}_L. \end{aligned}$$

where (a) follows from Lemma 4. \square

Lemma 7. Let \mathbf{y}_L and $\tilde{\mathbf{y}}_L$ be two sequences with type q . Let y_ℓ and \tilde{y}_ℓ be two sequences with type q and length ℓ . Then,

$$P(B(\mathbf{y}_L y_\ell, d) \cap B(\tilde{\mathbf{y}}_L \tilde{y}_\ell, d)) \leq P(B(\mathbf{y}_L, d) \cap B(\tilde{\mathbf{y}}_L, d)) \left(\frac{\mathbf{p}_{L+\ell}}{\mathbf{p}_L} \right)^2. \quad \square$$

Lemma 8. Let $N_{L,\mathcal{D}}(\mathbf{X})$ be the number of codelets of length L that match \mathbf{X} in codebook \mathcal{D}_L , and let $N_{L,\mathcal{D}}(\mathbf{X})$ be the number of codelets that match \mathbf{X} in a codebook \mathcal{D} . Then

$$\mathbb{E}N_{L+\ell,\mathcal{D}}^2 \leq (\mathbb{E}N_{L,\mathcal{D}}^2 + \mathbb{E}N_{L,\mathcal{D}}) \left(\frac{M_{L+\ell} \mathbf{p}_{L+\ell}}{M_L \mathbf{p}_L} \right)^2$$

where $\mathbf{p}_L = \mathbb{P}(B(y, d))$ for any $y \in T_q^L$. \square

For comparison let us consider the expected value of $\mathbb{E}N_{L,\mathcal{D}}^2$ for random codebook constructions of length L . Here we use codebooks \mathcal{C}_L populated with sequences of type q as follows. Generate independent sequences of length L , with the L -length sequence generated in step (i) being $\mathbf{X}^{(i)}$. Each $\mathbf{X}^{(i)}$ is in turn obtained by generating L bits *iid* Bernoulli $(1/2)$. Initialize $\mathcal{C}_L^{(0)} = \phi$. At every step i , update $\mathcal{C}^{(i)} = \mathcal{C}_L^{(i-1)} \cup \{y\}$, where y is a randomly chosen length L sequence of type q such that $\mathbf{X}^{(i)} \in B(y, D)$. Stop after the $i = M'_L$ th codelet is chosen, and let $\mathcal{C}_L = \mathcal{C}_L^{(M'_L)}$. For such a random codebook construction, it is easy to see that

$$\mathbb{P}(y' \in \mathcal{C}_L \text{ and } y \in \mathcal{C}_L) = \frac{M_L(M_L - 1)}{\mathcal{T}_L(\mathcal{T}_L - 1)}.$$

The above Lemmas imply

Corollary 9. $P(N_{L+\ell, \mathcal{D}} > 0) \geq \frac{P(N_{L, \mathcal{D}} > 0)}{P(N_{L, \mathcal{D}} > 0) + 1/(M_L \mathbf{p}_L)}$

Proof Cauchy Schwartz Inequality. □

The next cog in the proof is the observation that there cannot be too many “short” phrases in the lossy representation.

Lemma 10. For n sufficiently large, the number of nodes in the dictionary with length shorter than $\frac{\log n - 7\ell}{R(d)}$ is $\leq \frac{n}{(\log n)^2}$. □

The details of the reminder of the proof is omitted, but follows the following line of arguments standard in LZ analysis literature. (Complete below)

The section populated by short phrases contributes at most redundancy $1/(\log n)$. Unrolling Corollary 9, with high probability we find that some element of the dictionary matches an incoming phrase. Describing such phrases takes at most $\log n$ bits, and such phrases by Lemma 10 have length $\geq \log n - 7 \log \log n / R(D)$, yielding a per symbol encoding rate of $R(D) + \mathcal{O}(\frac{\log \log n}{\log n})$. With a small probability, no element of the dictionary matches an incoming phrase—forcing us to describe such phrases bit for bit, adding another $\mathcal{O}(\frac{\log \log n}{\log n})$ to the coding rate.

(Complete above)

5 Acknowledgments

We thank L. Lastras-Montaña for helpful discussions and constructive suggestions, as well as D. Baron, Y. Kochman, J. Østergaard, and G. Wornell for helpful discussions.

References

- [1] M. Kac. On the notion of recurrence in discrete stochastic processes. *Bulletin of the American Math Society*, 53:1002–1010, Oct 1947.
- [2] J. Ziv. On finite memory universal data compression and classification of individual sequences. *IEEE Transactions on Information Theory*, 54(4):1626–1636, 2008.
- [3] T. Weissman, E. Ordentlich, G. Seroussi, S. Verdu, and M. Weinberger. Universal discrete denoising: known channel. *IEEE Transactions on Information Theory*, 51(1):5–28, 2005. See also HP Labs Tech Report HPL-2003-29, Feb 2003.
- [4] A. Lempel and J. Ziv. On the complexity of finite sequences. *IEEE Transactions on Information Theory*, 22:75–81, 1976.
- [5] F. Kaspar and H. Schuster. Easily calculable measure for the complexity of spatiotemporal patterns. *Phys. Rev. A*, 36(2):842–848, Jul 1987.
- [6] N. Santhanam and D. Modha. Lossy lempel-ziv like compression algorithms for memoryless sources. In *Allerton Conference on Computing, Communication and Control*, September 2011.
- [7] T. Berger and J. D. Gibson. Lossy source coding. *IEEE Trans. Inform. Theory*, 44(6):2693–2723, 1998.
- [8] J. C. Kieffer. A survey of the theory of source coding with a fidelity criterion. *IEEE Trans. Inform. Theory*, 39(5):1473–1490, 1993.

- [9] Robert M. Gray and David L. Neuhoff. Quantization. *IEEE Trans. Inform. Theory*, 44(6):2325–2383, 1998.
- [10] M. J. Atallah, F. Chyzak, and P. Dumas. A randomized algorithm for approximate string matching. *Algorithmica*, 29:468–486, 2001.
- [11] G. Navarro. A guide to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.
- [12] K. Cheung and V. K. Wei. A locally adaptive source coding scheme. In *Bilkent Conf. on New Trends in Comm., Cont., and Signal Proc.*, pages 1473–1482, 1990.
- [13] En-Hui Yang and J. C. Kieffer. On the performance of data compression algorithms based upon string matching. *IEEE Trans. Inform. Theory*, 44:47–65, 1998.
- [14] Z. Zhang and V. K. Wei. An on-line universal lossy data compression algorithm via continuous codebook refinement—part i: Basic results. *IEEE Trans. Inform. Theory*, 42(3):803–821, 1996.
- [15] H. Morita and K. Kobayashi. An extension of LZW coding algorithm to source coding subject to a fidelity criterion. In *Proc. 4th Joint Swedish-Soviet Int. Workshop on Information Theory, Gotland, Sweden*, pages 105–109, 1989.
- [16] C. Constantinescu and J. A. Storer. On-line adaptive vector quantization with variable size codebook entries. In *Data Compression Conf.*, pages 32–41, 1993.
- [17] C. Constantinescu and J. A. Storer. Improved techniques for single-pass vector quantization. *Proceedings of the IEEE*, 82(6):933–939, 1994.
- [18] J. A. Storer. *Data Compression: Methods and Theory*. Computer Science Press, Rockville, Maryland, 1988.
- [19] Y. Steinberg and M. Gutman. An algorithm for source coding subject to a fidelity criterion, based on string matching. *IEEE Trans. Inform. Theory*, 39(3):877–886, 1993.
- [20] T. Luczak and W. Szpankowski. A suboptimal lossy data compression based on approximate pattern matching. *IEEE Trans. Inform. Theory*, 43:1439–1451, 1997.
- [21] I. Kontoyiannis. An implementable lossy version of the Lempel-Ziv algorithm—part i: Optimality for memoeyless sources. *IEEE Trans. Inform. Theory*, 45(7):2293–2305, 1999.
- [22] M. Atallah, Y. Genin, and W. Szpankowski. Pattern matching image compression: Algorithmic and empirical results. In *Proc. Int. Conf. Image Processing, Lausanne, Switzerland*, volume II, pages 349–352, 1996.
- [23] J. C. Kieffer and E.-H. Yang. Sequential codes, lossless compression of individual sequences, and Kolmogorov complexity. *IEEE Trans. Inform. Theory*, 42(1):29–39, 1996.
- [24] M. Alzina, W. Szpankowski, and A. Grama. 2D-pattern matching image and video compression. In *Data Compression Conf.*, pages 424–433, 1999.
- [25] R. Zamir and K. Rose. Towards lossy Lempel-Ziv: Natural type selection. In *Proc. Inform. Theory Workshop, Haifa, Israel*, page 58, 1996.

- [26] R. Zamir and K. Rose. Natural type selection in adaptive lossy compression. *IEEE Trans. Inform. Theory*, 47(1):99–111, 2001.
- [27] Y. Kochman and R. Zamir. Adaptive parametric vector quantization by natural type selection. In *Data Compression Conference*, pages 392–401, 2002.
- [28] En-Hui Yang and J. C. Kieffer. Simple universal lossy data compression schemes derived from the Lempel-Ziv algorithm. *IEEE Trans. Inform. Theory*, 42(1):239–245, 1996.
- [29] E.-H. Yang and Z. Zhang. Variable-rate trellis source encoding. *IEEE Trans. Inform. Theory*, 45(2):586–608, 1999.
- [30] Y. Matsunaga and H. Yamamoto. A coding theorem for lossy data compression by LDPC codes. *IEEE Trans. Info. Theory*, 49:2225–2229, 2003.
- [31] M. J. Wainwright and E. Maneva. Lossy source coding by message-passing and decimation over generalized codewords of LDGM codes. In *International Symposium on Information Theory*, Adelaide, Australia, September 2005. Available at arxiv:cs.IT/0508068.
- [32] E. Martinian and M. J. Wainwright. Low density codes achieve the rate-distortion bound. In *Data Compression Conference*, volume 1, pages 153–162, March 2006. Available at arxiv:cs.IT/061123.
- [33] A. G. Dimakis, M. J. Wainwright, and K. Ramchandran. Lower bounds on the rate-distortion function of LDGM codes. In *Information Theory Workshop*, September 2007.
- [34] S. Jalal, A. Montanari, and T. Weissman. Lossy compression of discrete sources via viterbi algorithm, 2010. arXiv:1011.3761v2 [cs.IT] 21 Nov 2010.
- [35] D. S. Modha. Codelet Parsing: Quadratic-time, sequential, adaptive algorithms for lossy compression. In *Proc. DCC, Snowbird, UT*, March 24–27, 2003.
- [36] D. S. Modha. The art of making mistakes: A quadratic-time, sequential, adaptive algorithm for lossy compression. Technical Report RJ 10286, IBM Almaden Research Center, San Jose, CA, February 19, 2003.
- [37] D. Modha and N.P. Santhanam. Making the correct mistakes. In *Proceedings of the Data Compression Conference*, 2006.
- [38] A. Dvoretzky and Th. Motzkin. A problem of arrangements. *Duke Mathematics Journal*, 14:305–313, 1947.
- [39] M. Renault. Four proofs of the ballot theorem. *Mathematics magazine*, 80(5), December 2007.
- [40] N. Cesa Bianchi and G. Lugosi. *Prediction, Learning and Games*. Cambridge University Press, 2006.